

A J2ME Client for the CSU, Chico Portal

Alex Petzinger, California State University, Chico

Abstract— with the conversion of the Chico Portal to a pure J2EE implementation (uPortal), a J2ME client-component would provide a natural extension. Such an addition could extend access to the Portal to a multitude of Java-based mobile devices without sacrificing security and functionality.



1 INTRODUCTION

Enterprise mobility is one of the most promising and cutting-edge technologies in the IT industry . As the creation and use of IT solutions grows with the spread of the Internet, almost all of today’s users are still limited to simply pen- and paper-based solutions when out in the field because of a lack of network coverage, lack of a typical PC-based device, or both. As wireless Internet access become more ubiquitous, companies will rely more heavily on mobility extensions to their existing IT solutions and thus cut down on their “in the field” bottle-neck and increase efficiency. In order to meet this demand, an explosion of small-device Java solutions is happening now. Java™ 2 Platform, Micro Edition (J2ME) is important for two reasons. First, it provides a common software platform that can be used across a wide variety of small devices. Second, J2ME has important built-in security features lacking in other wireless protocols. For these reasons, J2ME continue to advance in the enterprise IT market just as J2EE (Java 2 Enterprise Edition) had expanded as the prevailing standard on the server side a decade earlier.

As part of these ongoing trends, I am proposing to develop a J2ME solution for the CSU, Chico Portal that will address several issues. According to the Director of Applications Development and Enterprise Design for the Chico Campus, Debra McElroberts, the current implementation of the Portal will be discarded for an open source implementation known as uPortal. This newer version of the

Chico Portal is scheduled for deployment by the end of 2005. As stated at the uPortal home page, “uPortal is an open-standard effort using Java, XML, JSP and J2EE.” [1]. A J2ME client would therefore provide a natural extension for the implementation of uPortal here at the Chico State Campus. However, a full-scale J2ME solution that integrated completely with the actual Chico portal would be a daunting task. Such an undertaking would include full integration with the entire campus-wide Oracle/Peoplesoft infrastructure. Besides the time devoted towards its development and integration, such an undertaking would necessarily include rigorous testing of its functionality, completeness, and security. However, for the purposes of this project, a “stripped-down” architecture demonstrating the feasibility of a J2ME-campus portal would be a reasonable endeavor. As such, this project will be implemented using a three-tier architecture that will closely mimic the Chico portal using as much of the same software and technologies as possible:

sible:

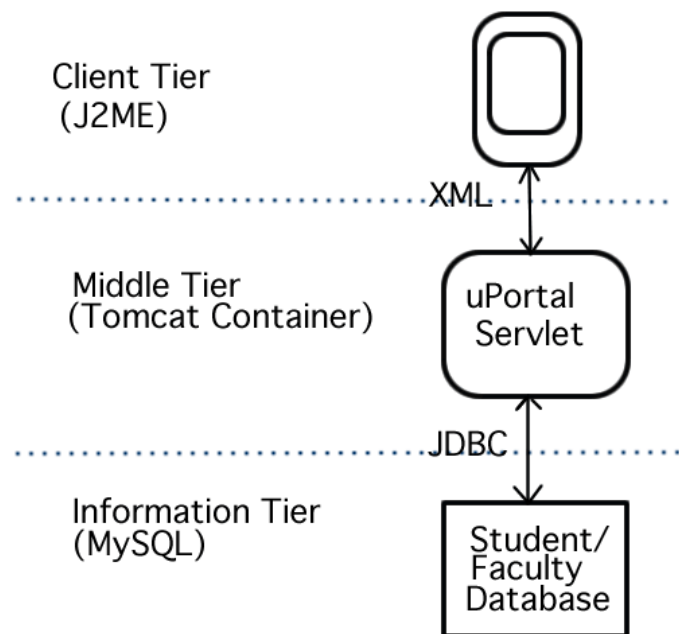


Figure 1

2 PRELIMINARY INVESTIGATION

Based on earlier work with enterprise J2ME a J2ME application for the proposed uPortal implementation at Chico would be both very instructive and useful. Since Sun Microsystems first introduced J2ME in 1999, the literature specifically targeting J2ME enterprise development has been steadily growing. Besides the abundant tutorials, specifications, and software tools that are available online from Sun and other vendors, several books have appeared specifically addressing use of J2ME with enter-

prise applications. Indeed, there have been earlier projects demonstrating the use of J2ME with campus portals [2]. One of my wireless Java reference books describes in detail a campus portal for wireless devices that was a project conceived at San Jose State University. This project was the effort of several students completing their master's projects in 2001[3]. However, these efforts used a less robust version of the J2ME standard (MIDP 1.0) I will be using a significantly more enhanced version of J2ME built upon MIDP 2.0 (Mobile Information Device Protocol). J2ME applications are hereafter referred to as "MIDlets". The newer MIDP 2.0 has substantial improvements over the earlier 1.0 standard including an enhanced user interface, media support, game support, push architecture, over-the-air (OTA) provisioning, and end-to-end security [4]. For the purposes of the J2ME MIDlet portal, this has several significant implications: the user interface improvements make MIDlets more interactive and easier to use. Developers have more control over the application's layout, screen size, and UI components. This increases portability over a wider variety of devices than before since each device's MIDP implementation optimally controls the screen size and layout for the application. MIDP 1.0 only provided support for HTTP, but MIDP 2.0 adds support for leading connectivity standards beyond HTTP, such as secure HTTP (HTTPS), datagram, sockets, server sockets, and serial port communication. Support for HTTPS is vital for any mobile portal adaptation since almost all campus portals require HTTPS (including the current and future Chico implementations). Beside HTTPS support, MIDP 2.0 also leverages existing standards such as SSL (secure socket layer) and WTLS Wireless Transport Layer Security. WTLS is the security layer of the WAP (wireless application protocol), to enable the transmission of encrypted data. Much as with J2ME, WAP is also a secure specification that allows users to access information instantly via handheld wireless devices. But, as

Sun Microsystems has noted, WAP should be regarded as a complementary standard to J2ME and not a competing one [5]. Strictly speaking, WAP is a browsing capability that uses its own markup language (WML) geared for small devices, particularly cell phones. Nevertheless, the main players that initiated the WAP Forum (Nokia, Ericsson, and Motorola) also actively support J2ME in their latest products. WAP 2.0 (the latest release) still requires a proxy server (known as a WAP gateway) to handle protocol conversions between Internet-oriented protocols (TCP/IP, SSL, etc.) and the wireless network [6]. For the developer, this means significantly more coding (including WMLscript) and testing. In terms of complexity and time constraints, a pure WAP-based campus portal would qualify in its own right as a unique master's project. As mentioned before, MIDP 2.0 provides end-to-end security over TCP/IP without the need of a gateway to do protocol conversion. The wireless downloading of MIDlets is known as

over-the-air (OTA) provisioning. This allows a consumer to find an application on a server, download it to their phone or other wireless device where it is automatically installed. This feature in MIDP 2.0 dictates that provisioning is implemented over HTTP 1.1 as well as over WAP [7].

Together with its MIDP layer, J2ME address more than just mobile phones, pagers, and personal digital assistants. All these resource-constrained devices use a particular base set of application programming interfaces and a virtual machine known as CLDC (Connected Limited Device Configuration). The Connected Device Configuration (CDC) was defined for more capable (that is, more memory, better display, and so on) devices such as TV set-top boxes, automobile navigation system, and gaming systems. As a matter of course, my project will be focused on the most commonly used devices which are all based on the smaller CLDC platform. The following diagram illustrates the relationship between

J2ME, its configuration and profiles, and J2EE, and J2SE:

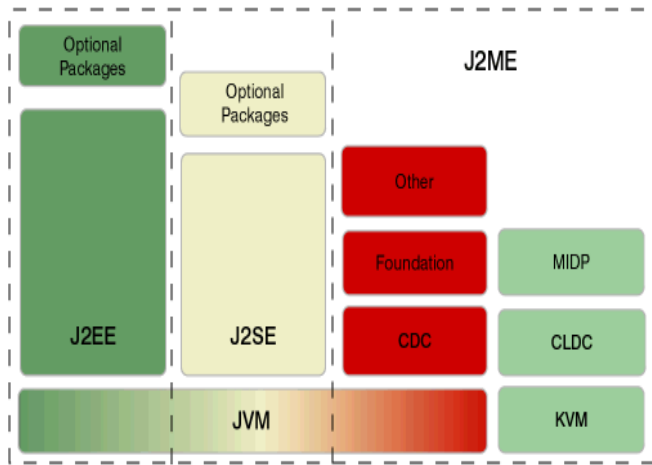


Figure 2 (courtesy of IBM)

But just as Java has increasingly come to represent mobile code across a variety of computing devices, XML has come to represent mobile data. As a result, several XML parsers exist for the J2ME platform with kXML being the most popular and best-supported [8]. A J2ME client could also manipulate plain text, but for incorporation with the uPortal, support for XML is required. Whatever client application is used, uPortal relies completely on XML and XSLT (Extensible Stylesheet Language Transformations) to create the final user layout. The uPortal middle-tier interacts with the

J2ME client as a Java Servlet within an Apache Tomcat application server. As the layer between both the client and information tiers, uPortal implements the bulk of the business logic.

Database tables represent the 3rd tier (information tier) of our J2ME portal project. There are several different schemes that allow a J2ME MIDlet to access remote databases [9]. Java Database Connectivity (JDBC) provides the Java API that can interact with any SQL compliant database. This includes using J2ME JDBC implementations that allow a direct connection to a remote database connection. One approach is the Oracle J2ME SQL SDK which enables mobile clients to access backend databases through the Oracle9i Mobile Application Server. Another method is use of a gateway servlet that takes requests the MIDlet client, queries any SQL-compliant database (via JDBC or other SQL driver), and sends this back to the MIDlet client via HTTP. For almost any existing enterprise application, there are a plethora

of commercial and open source SDK's and toolkits specific for J2ME/J2EE development for both new and legacy applications.

3 PROPOSED METHOD

As part of my extensive research into J2ME, I've created some sample MIDlets using sample code with a variety of both free and commercial tools. My approach will be to test and compare these tools and see how well they accomplish my goals. WebSphere Studio Device Developer from IBM includes several components and IDE for operability with web services and is available as a trial download. Simplicity for Mobile Devices and Simplicity Enterprise are RAD (Rapid Application Development) IDE that allow one to build gateway servlets and handle micro XML processing. Simplicity Enterprise includes its own built-in HTTP server and servlet engine for both J2ME and J2EE development. Together with its PointBase embedded database, one can easily create end-to-end solutions. I will need to explore more of the many capabilities of the Sim-

licity Suite and how it can integrate with uPortal. Yet another tool is the Oracle9i JDeveloper which, similar to the other tools mentioned, is seamlessly integrated the J2ME technology. As part of my previous work with Java and servlets, I manage my own Apache2 web server with a Tomcat 4.1 servlet container. Running on BSD UNIX on a Macintosh G4 (Mac OS X 10.3.8), this platform has all the requirements necessary to operate and configure uPortal. Further investigation is needed to see which database I should use with the uPortal servlet. Depending upon the database capabilities of my J2ME tools, I can use Oracle 9i, or MySQL. After installation of uPortal, I plan to create a desktop client that accesses the backend database through uPortal. Further development of this could include a Javabean or EJB that handles SQL requests. Proceeding from the other end, I will create a simple MIDlet using Simplicity suite (or similar tool) and create a prototype that performs simple database transactions. Testing MIDlets is typically per-

formed with an onscreen emulator before it is downloaded to the mobile device.

XML parsing from the servlet to the MIDlet device is another important issue. Further investigation will reveal if I can use the proprietary XML parser used by the Simplicity suite, or if I can integrate the aforementioned kXML parser to my development. Other design decisions will consider use of an XML Encryption API or configuring my application server to use HTTPS (SSL) for security. Another issue is size of the JAR (Java Archive) that is generated by the J2ME tools. The open source Bouncy Castle SDK includes an obfuscator which decreases the size of the JAR file. Downloaded to a memory-constrained device, this increases runtime efficiency. Finally, I must address over-the-air provisioning for my MIDlet. This requires configuring my application server to handle the J2ME MIME types (Multipurpose Internet Mail Extensions) along with changes to the MIDlet. Altogether, there is extensive testing to accomplish. My strategy

is to create a simple prototype that can accomplish all these tasks described above, and then add more functionality to the MIDlet as I increase tables to the backend database. The guiding philosophy here is to mimic as much functionality of a real portal. In all, this will demonstrate the feasibility of J2ME with a student portal, a sure “proof of concept”.

PROPOSED SCHEDULE

COMPLETION DATES:

1. Implementation and testing of uPortal on Apache server (Feb. 30, 2006).
2. Addition of an HTML web client to uPortal (March 10).
3. Development of a JavaBean or Enterprise JB for JDBC access to the portal (March 31).
4. Testing and comparison of various J2ME tools: Oracle, Data Representations (Simplicity), IBM Websphere, the Sun Wireless Toolkit (April 25).
5. Development of a J2ME MIDlet with remote database access (May 20).
6. Testing of an actual MIDlet (non-emulation) on a wireless device (June 15).
7. Integration and testing of various database components and JDBC (June 30).
8. Selection and testing of a suitable XML parser between server and client (July 31).

9. Use of either XML encryption or SSL server configuration for security (August 15).
10. Further testing of the MIDlet on a wireless device (August 31).
11. Use of an obfuscator (Sept. 10).
12. Configuration and testing of OTA provisioning to a wireless device (Sept. 30).
13. Additional functionality and testing as time permits. (October 20).

REFERENCES

- [1] <http://www.fair-portal.hull.ac.uk/downloads/uPortalGuide1.pdf>
Chamberlain, Lee, Juahkah, Margaret, Sherratt, David
The Java Architectures Special Interest Group (JA-SIG)
July 4, 2003
- [2] <http://cslab.engr.sjsu.edu/~dharkey/TJ.pdf>
Setiawan, Tjahjadinata, Nov, 2001.
- [3] Harkey, Dan, Wireless Java Programming for Enterprise Applications, Wiley Publishing, pp. 601-653, 2001
- [4] <http://java.sun.com/products/midp/endtoend>
Sun Microsystems, Inc. 1994-2004
- [5] <http://developers.sun.com/techtoc/mobility/midp/articles/midpwap/>
Mahmoud, Qusay H. Sun Microsystems, Inc.
February 2002
- [6] <http://java.sun.com/features/2000/07/wireless.html>
Hardee, Martin Sun Microsystems, Inc.
Copyright 1994-2004
- [7] http://www.ftponline.com/javapro/2003_03/magazine/features/dhemphill
Hemphill, David & . Morehouse. David
JavaPro Magazine, March 2003
- [8] <http://www.devx.com/xml/Article/11773>
Robert Cadena, Jupitermedia Corporation,
Copyright 2004
- [9] Yuan, Michael, Enterprise J2ME, Upper Saddle River, New Jersey, Prentice Hall, pp. 272-274, 2004.
- [10] <http://www.oracle.com/technology/tech/wireless/tools/j2me/index.html> Copyright © 2005, Oracle.